DOCUMENT RESUME

ED 179 086 FL 010 583

AUTHOR Ornan, Uzzi
TITLE Generating and Transforming by a Computer without a Dictionary.
PUB DATE 78
NOTE 13p.
AVAILABLE FROM University of Cambridge, Literary and Linguistic Computing Centre, Sidgwick Site, Cambridge CB3 9DA, England
JOURNAL CIT Association for Literary and Linguistic Computing Bulletin: v6 n3 p280-91 1978

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Computational Linguistics; *Computer Assisted Instruction; Computer Programs; Educational Technology; *Generative Grammar; Grammar; *Hebrew; Information Processing; Language Patterns; *Language Processing; Linguistic Theory; Machine Translation; *Programing Languages; Second Language Learning; Sentence Structure; Syntax; Verbs

ABSTRACT

The ability of the computer to generate output not included in the input may be used for linguistic as well as for computational input. The ability to accept linguistic data and process it according to a certain program seems to be a promising field for investigation. Progress in this field may strengthen the assumption that the computer can be considered as a model for at least that part of the human brain that contains "knowledge of language(s)." If, for example, a program can be written for all inflected forms (in an inflectional language), when the input is an entry-word, the program will be stronger than any published conjugation chart. In programs for generating or transforming sentences, well-built sentences that have never before been heard or written may be generated without any restriction. Students may explore the field for themselves in an inductive way with the computer acting as a reliable "informer." As an illustration of the preceding, two examples are given for the generating ability of the computer: inflection Hebrew verbs and the extra-position transformation of Hebrew sentences. An explanation is given for each example as well as charts printed by the computer. Programs such as these might be useful for students of foreign languages. It is suggested that they might also show how far a computer can help to formulate grammar. (AMH)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION
THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

GENERATING AND TRANSFORMING BY A COMPUTER WITHOUT A DICTIONARY

Uzzi Ornan (Hebrew University of Jerusalem)

* *

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY
Mrs J. M. Smith
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

1. TWO USES OF THE COMPUTER

The most popular use of computers is arranging items according to the wish of the user, or selecting certain items from a huge store. The items do not change. The output contains the same items, or part of them, arranged differently.

The other use of the computer is its computational use, such as the ability to multiply two numbers. The output is a new item, which was not included in the input. This use of the computer relies on the generating ability of the computer.

2. COMPUTER AS A MODEL

This generating ability of the computer may be used for linguistic input also. The ability to accept linguistic data and process it according to a certain program seems to be a very promising field for investigation. Progress in this field may strengthen the assumption that the computer can be considered as a model for at least that part of the human brain that contains 'knowledge of language(s)'.

More specifically, we may assume that the memory contains lexical entries and idioms, but does not contain inflecteed forms of entries nor concatenations of words into clauses or sentences. In another location programs are stored. These programs are grammatical rules. Some of them may generate conjugational and declensional forms, others generate clauses and sentences. All rules specify the conditions under which they should or may be in force. There should also exist semantic and socio-linguistic mechanisms where results of the generating rules are checked in order to 'abolish' or reshape the output.

3. COMPUTER PROGRAMS AS A GENERATIVE GRAMMAR

If we can write a program for all inflected forms (in an inflectional language) when the input is any entry-word - this program will be 'stronger' than any published 'Conjugation Chart'. A conjugation chart[1] deals with words that exist in the language, while a program may generate inflected forms of all possible words, including potential words which have not yet been coined.

Besides, every conjugation chart must have certain examples chosen by the author, which, by their very nature, are preferred to all other cases. A program, on the other hand, does not include any example, but produces it at your request. Thus, your choice is preferred.

The same applies to programs for generating or transforming sentences. Well-built sentences that have never been heard or written before may be generated without any restriction.

This feature of generating well-built inflected forms and sentences seems to make these programs applicable for Computer Assisted Instruction. By 'playing' with his terminal the student of a foreign language is exposed to a reliable 'informer' whom he may ask all questions in certain grammatical fields. The programs will

FL010583

always print right answers for him. Students may explore the field for themselves in an extremely inductive way; they may take a little or much advice from their instructor concerning the material for their 'conversation' with the computer; they may consult their handbook occasionally or all along the way - everyone as he finds it best for himself, or everyone as he finds it best for his students.

I will try to specify here two examples for the generating ability of the computer: inflecting Hebrew verbs, and the extra-position transformation of Hebrew sentences.

ED179086

4. SHORT INTRODUCTION TO HEBREW MORPHOLOGY[2]

Dictionary words (=entries) in Hebrew stem from the following sources:
(a) words which are generated in Hebrew itself, in a mechanism which we will term 'Root-Pattern Array' (see below, 5.);
(b) Ancient words taken from a proto-semitic vocabulary;
(c) Loan words from various languages.

Apart from these three sources, there are also
(d) dictionary words which are combinations of other dictionary words (sometimes these combinations are shortened or abbreviated), and also
(e) words which consist of a combination of a dictionary word and an affix. It should be noted that this affix is usually not a word, but a morpheme used as a grammatical device. (e.g. 'on' as a suffix is used to signify a referent which is smaller or 'not real', comparing with the referent of the word without this suffix: z'ev (wolf) - z'evon (young or small, or would-like-to-be, wolf).

Hebrew is an inflected language with many inflected forms. Especially, Hebrew verbs in all their tenses have affixes which are attached pronouns denoting the actor; Hebrew nouns have also suffixes as attached pronouns which denote the possessor of the referent of the noun, etc..

Inflected-forms in Hebrew are produced as a combination of a dictionary-word with an affix (either suffix or prefix). The affix serves as a pronoun, and when it is a verb, a different affix serves for each tense, hence affixes signify the tense also.

It should be noted that these combinations, as well as words from the first three sources, must go through a mechanism of realization where phonetic rules or morphophonetic rules are applicable before they are ready for use.

An abbreviated summary of Hebrew morphology may be seen in Figure A.

5. ROOT-PATTERN ARRAY

I would like to say a few more words on what we called above 'Root-Pattern Array'. Every Hebrew word of this source can be thought of as existing in a certain square of a huge array. Each row of this array is indicated by a series of three or four consonants (the 'root') which appears at the left end of the row; each column is indicated by a pattern at its head. The pattern consists of vowels only, or vowels with a consonant or two, while three or four empty slots are dispersed in between. Examples for patterns: The word gamar has the pattern -a-a-; the pattern of minzar is mi--a; of tilbošet is ti--o-et; of šipper is -i--e-, etc..

Each square of the array is located in a certain column and a certain row. Thus, a word may be generated in this square when the consonants of the root are put into the empty slots of the pattern. Thus, root GMR and pattern -a-a- generate the above mentioned gamar, NZR and mi--a- make minzar, etc..

It should be noted, however, that many squares in the array do not contain any word. These squares preserve 'potential words' which do not exist in the language. They become useful when speakers search for accurate expression and feel that a new word is needed.

Circle No. 1 in Figure A receives its words from the root-pattern array, which may be described as a mechanism for generating words, as shown in Figure B.

Morphological description of this kind enables the linguist to write a program. It should be noted that the program does not contain any of the lists mentioned in 1, 3, 5, or 11 in Figure A.

The program does contain, however, the rules which act in the areas 2, 4, 6, 9, 10, 12, and 13. Also, the program should contain lists of affixes in 14 and 17, and, of course, the rules which act in 15, 16, 18, and 19.

The words in circle 1 exist in the dictionary, but, as explained above, a new word of this kind may appear, even though it is not included in the dictionary. This fact shows how a grammar as a program is stronger than any list (as mentioned above, 3). We should add here that there are certain formal ways of adding new roots to the list (Figure B).[3] Also, new patterns have occasionally been shaped in Hebrew.

6. GENERATING INFLECTED FORMS

Generating in the areas 18 and 19 with results accepted in 21 (through 20) proves to be very interesting. If we become used to this kind of grammar, the process is much shorter. Instead of 'Conjugation Charts' of dozens of pages, only a few flow charts are needed.

I would like to present here an algorithm of one progam from those which I have compiled. It generates inflected forms of the Past Tense. I should begin by saying that the entry of a verb in a Hebrew dictionary is usually the basic form of Past Tense. This form is also the theoretical form of the 3rd person masculine singular, in the past. In other words, 3rd p.m.s. takes suffix Ø. The full list of Past Tense suffixes are the following: -ti, -ta, -t, -Ø, -a, -nu, -tem, -ten, -u. See Figure C. Several examples of the output are given in 7. The program is written in PASCAL.

The reader might like to check whether he can get correct results by using the flow chart of Figure C. In the top rectangle he should place an item of input, taken from the examples of 7. He should follow its way according to the questions and instructions. His results should then be compared with the correct answer printed in the examples in 7. The reader who knows Hebrew might try other verbs. He will probably find out that the chart correctly generates all Hebrew verbs.

7. EXAMPLES OF VERB INFLECTION OF PAST TENSE

What follows is a chart printed by the computer. It is divided into three columns for masculine, common, and feminine forms. 'Common' means here forms which are used for both m. and f.. Hebrew does not have special forms for 'it'.

Horizontally, the chart is divided into singular and plural, each of them contains all three persons, which are indicated by the numbers 1, 2 and 3.

Letters for Hebrew phonemes used here are as follows:

vBGDHWZX♦YKLMNSAPCQR$TAEIOU. V and F are excluded, as they do not form separate phonemes, but are variants, rather, of B and P, respectively. The phonetic values of Hebrew phonemes are similar to those of English except in what follows: /w/ - [v], /x/ - Scottish [ch], /♦/ → [t], /c/ → [ts], /$/ → [sh], /v/ and /^/ are guttural sounds which you may omit; ([v] is a glottal stop, [^] is a voiced pharyngeal fricative; X and H are gutturals too). Vowels are pronounced as in Italian.

```
XALA
INPUT: -
INFLECTED FORMS ARE:-
                          X A L A

                    MASC.                    FEM.
                    -----                    ----
SING.(1):                        XALITI
-----(2):           XALITA                    XALIT
     (3):           XALA                      XALTA
PLUR.(1):                        XALINU
-----(2):           XALITEM                   XALITEN
     (3):                        XALU

HIPLIv PLEASE NOTE THAT F SHOULD BE REPLACED BY P.
INPUT: -
INFLECTED FORMS ARE:-
                          H I P L I v

                    MASC.                    FEM.
                    -----                    ----
SING.(1):                        HIPLEvTI
-----(2):           HIPLEvTA                  HIPLEvT
     (3):           HIPLIv                    HIPLIvA
PLUR.(1):                        HIPLEvNU
-----(2):           HIPLEvTEM                 HIPLEvTEN
     (3):                        HIPLIvU

NAVAX PLEASE NOTE THAT V SHOULD BE REPLACED BY B.
INPUT: -
INFLECTED FORMS APE:-
                          N A B A X

                    MASC.                    FEM.
                    -----                    ----
SING.(1):                        NABAXTI
-----(2):           NABAXTA                   NABAXT
     (3):           NABAX                     NABXA
PLUR.(1):                        NABAXNU
-----(2):           NBAXTEM                   NBAXTEN
     (3):                        NABXU

(THE CLASSICAL FORM OF:          NABAXT   IS:-          NABAXAT)

LAXA$
INPUT: -
INFLECTED FORMS ARE:-
                          L A X A $

                    MASC.                    FEM.
                    -----                    ----
SING.(1):                        LAXASTI
-----(2):           LAXA$TA                   LAXA$T
     (3):           LAXA$                     LAXA$A
PLUR.(1):                        LAXASNU
-----(2):           LXASTEM                   LXASTEN
     (3):                        LAXASU

HEQIM
INPUT: -
INFLECTED FORMS APE:-
                          H E Q I M
```

		MASC.		FEM.
SING.	(1)		HEQAMTI	
	(2)	HEQAMTA		HEQAMT
	(3)	HEQIM		HEQIMA
PLUR.	(1)		HEQAMNU	
	(2)	HAQAMTEM		HAQAMTEN
	(3)		HEQIMU	

ANOTHER POSSIBILITY IS:-

SING.	(1)		HAQIMOTI	
	(2)	HAQIMOTA		HAQIMOT
	(3)	HEQIM		HEQIMA
PLUR.	(1)		HAQIMONU	
	(2)	HAQIMOTEM		HAQIMOTEN
	(3)		HEQIMU	

ALL YOU WANTED TO KNOW AND DARED TO ASK HAS BEEN PRINTED HERE...

8. TRANSFORMATIONS IN SYNTAX

The computer can generate sentences and apply transformations on given sentences as well. Here also no lexical list is needed. Every well-built sentence which is given as an input will be treated by the program and will be changed according to grammatical rules into a new sentence with a different structure.

I have written a program for transforming sentences into extra-position structure.[4] Each noun in any sentence may be taken from the sentence and put before it, provided that several conditions are fulfilled:

(a) a pronoun should replace this noun in its previous place;
(b) this pronoun should combine with the preposition which precedes it or join with the direct object marker whether it exists or not;
(c) certain rules relating to the word-order should take effect.

The matter becomes more complicated as agreement of gender and number in nouns, pronouns, verbs, and adjectives should be preserved. The program, written in SNOBOL 4, should look first for formal markers of gender and number, then it may choose correct steps. This program has an interactive version, and it acts as follows:

First the user is asked to input a sentence at the terminal. The program repeats it (without mistakes, if there were any), then it computes and prints clear cases of extra-position versions of the given sentence. Now the user is asked for further details (the questions are always formulated as:- 'Do you mean "........"?'). The answer should be either 'Yes' or 'No'. According to the answer more extra-position sentences are printed out, or the program decides that no other extra-position sentences are possible. When this happens 'Your next sentence, please?' will appear.

The sentences for the examples are the following (with their 'pidgin' translations):

(1) (2) 'ANAXNU NA&TIQ 'ET HA-ŠIR LA-MAXBERET B-&IPPARON;[5]
we'll copy[6] the-poem into-the-notebook with-pencil.

(3) YLADIM TAPSU ŠAM XATULIM;
children caught there cats.

(4) MASARTI MIKTAB LA-MNAHEL.
I delivered letter to-the-headmaster.

(5) NASA&TI 'ETMOL LA-&IR.
I went (by a vehicle) yesterday to-the-town.

9. PROCESSED SENTENCES IN INTERACTIVE MODE

These sentences are processed by the program as follows (capital letters are written by the user of the terminal; small letters are written by the computer):

```
your sentence, please?
ANU NA&TIQ 'ET HA-ŠIR LA-MAXBERET.                    'Anu = 'Anaxnu
sent.  1 is: - 'anu na&tiq 'et ha-šir la-maxberet.

result 1 is: -
ha-šir - 'anu na&tiq 'oto la-maxberet.
do you mean "'anu na&tiq 'et ha-šir el ha-maxberet"?
YES
result 2 is: -
ha-maxberet - 'anu na&tiq 'eleiha et ha-šir.
your next sentence, please?
'ANAXNU NA&TIQ 'ET HA-ŠIR LA-MAXBERET B-&IPPARON.
sent.  2 is:- 'anaxnu na&tiq 'et ha-šir la-maxberet b-&ipparon.
result 1 is:-
ha-šir - 'anaxnu na&tiq 'oto la-maxberet b-&ipparon.
do you mean "anaxnu na&tiq 'et ha-šir 'el ha-maxberet b-&ipparon"?
YES
result 2 is:-
ha-maxberet - a'anaxnu na&tiq 'eleiha 'et ha-sir b-&ipparon.
result 3 is:-
h   pparon - 'anaxnu na&tiq bo 'et ha-šir la maxberet.
your next sentence, please?
YLADIM TAPSU ŠAM XATULIM.
sent.  3 is:-
yladim tapsu šam xatulim.
do you mean "'et ha-yladim tapsu šam xatulim"?
NO
do you mean "yladim tapsu šam 'et ha-xatulim"?
YES
result 1 is:-
ha-xatulim - yladim tapsu 'otam šam.
your next sentence, please?
MASARTI MIKTAB LA-MNAHEL.
sent.  4 is:- masarti miktab la-mnahel.
do you mean "masarti miktab 'el ha-mnahel"?
NO
result 1 is:-
ha-mnahel - masarti lo miktab.
do you mean "masarti 'et ha-miktab la-mnahel"?
YES
result 2 is:-
ha-miktab - masarti 'oto la-mnahel.
your next sentence, please?
NASA&TI 'ETMOL LA-&IR.
sent.5 is: - nasa&ti 'etmol la-&ir.
do you mean "nasa&ti 'etmol 'el ha-&ir"?
YES
result 1 is:-
ha-&ir - nasa&ti 'eleiha 'etmol.
do you mean 'nasa&ti 'et ha-'etmol la-&ir'?
NO
your next sentence, please?
```

10. PROCESSED SENTENCES IN BATCH MODE

The same program can be adjusted for use in batch mode when all sentences to be processed are given as data. The following is such a program. Here = has been added in several cases to enable the input to be converted into Hebrew characters according to another program on which I reported in Hebrew in 1976.[7]

```
NO ERRORS DETECTED IN SOURCE PROGRAM


   SENTENCE NO. 1 IS: - ~ANU NA^TIQ ~ET HA-$IR LA-MAXBERET.
   RESULT NO. 1 IS: -
HA-$IR - ~ANU NA^TIQ ~OTO LA-MAXBERET.

   IF YOU MEAN: ~ANU NA^TIQ ~ET HA-$IR ~EL HA-MAXBERET.,
   THEN -
   RESULT NO. 2 IS: -
HA-MAXBERET - ~ANU NA^TIQ ~ELEIHA= ~ET HA-$IR.

ELSE,
   RESULT NO. 2 IS: -
HA-MAXBERET - ~ANU NA^TIQ LAH ~ET HA-$IR.


   SENTENCE NO. 2 IS: - ~ANAXNU NA^TIQ ~ET HA-$IR LA-MAXBERET B-^IPPARON.
   RESULT NO. 1 IS: -
HA-$IR - ~ANAXNU NA^TIQ ~OTO LA-MAXBERET B-^IPPARON.

   IF YOU MEAN: ~ANAXNU NA^TIQ ~ET HA-$IR ~EL HA-MAXBERET B-^IPPARON.,
   THEN -
   RESULT NO. 2 IS: -
HA-MAXBERET - ~ANAXNU NA^TIQ ~ELEIHA= ~ET HA-$IR B-^IPPARON.

ELSE,
   RESULT NO. 2 IS: -
HA-MAXBERET - ~ANAXNU NA^TIQ LAH ~ET HA-$IR B-^IPPARON.

   RESULT NO. 3 IS: -
HA-^IPPARON - ~ANAXNU NA^TIQ BO ~ET HA-$IR LA-MAXBERET.


   SENTENCE NO. 3 IS: - YLADIM TAPSU $AM XATULIM.
   IF YOU MEAN: ~ET HA-YLADIM TAPSU $AM XATULIM.,
   THEN -
   RESULT NO. 1 IS: -
HA-YLADIM - TAPSU ~OTAM $AM XATULIM.

   IF YOU MEAN: YLADIM TAPSU $AM ~ET HA-XATULIM.,
   THEN -
   RESULT NO. 2 IS: -
HA-XATULIM - YLADIM TAPSU ~OTAM $AM.


   SENTENCE NO. 4 IS: - MASARTI MIKTAB LA-MNAHEL.
   IF YOU MEAN: MASARTI MIKTAB ~EL HA-MNAHEL.,
   THEN -
   RESULT NO. 1 IS: -
HA-MNAHEL - MASARTI ~ELAW MIKTAB.

ELSE,
   RESULT NO. 1 IS: -
HA-MNAHEL - MASARTI LO MIKTAB.

   IF YOU MEAN: MASARTI ~ET HA-MIKTAB LA-MNAHEL.,
   THEN -
   RESULT NO. 2 IS: -
HA-MIKTAB - MASARTI ~OTO LA-MNAHEL.
```

```
   SENTENCE NO. 2 IS: - NASA^TI ^ETMOL LA-^IR.
   IF YOU MEAN: NASA^TI ^ETMOL ^EL HA-^IR.,
   THEN -
   RESULT NO. 1 IS: -
HA-^IR - NASA^TI ^ELEIHA- ^ETMOL.

ELSE.
   RESULT NO. 1 IS: -
HA-^IR - NASA^TI LAH ^ETMOL.

   IF YOU MEAN: NASA^TI ^ET HA-^ETMOL LA-^IR.,
   THEN -
   RESULT NO. 2 IS: -
HA-^ETMOL - NASA^TI ^OTO LA-^IR.


NORMAL TERMINATION
LAST STATEMENT EXECUTED WAS   23


SNOBOL4 (VERSION 3.8)
STATISTICS SUMMARY:

        2974 MS. COMPILATION TIME
        9240 MS. EXECUTION TIME
         973 STATEMENTS EXECUTED,        311 FAILED
          24 ARITHMETIC OPERATIONS PERFORMED
         504 PATTERN MATCHES PERFORMED
           9 REGENERATIONS OF DYNAMIC STORAGE
           6 READS PERFORMED
          76 WRITES PERFORMED
        9.49 MS. AVERAGE PER STATEMENT EXECUTED

 //  HBBDJDZ. LTD6  FC          508 LINES          8 PAGES END OF LIST //
```

11. CONCLUSION

As mentioned above, one might find such programs useful for students of foreign languages. You simulate here a 'Native speaker' who would answer patiently every question which may have been raised by the student. 'He' will also comment on several common mistakes (only a few of them have been shown here).

I would like to add, however, that programs like these might show also how far a computer can help formulate grammar. It seems to me that computers are the best device for a linguist for verifying the rules that he formulates. The reason for this is that in formulating grammar as a program the linguist cannot use such terms as 'generally', or 'usually'. Also, he is prevented from using any 'etc.' which is so common in handbooks for grammar. Most important is that no semantic consideration should be involved.

No Hebrew verb, for instance, should be manipulated on the basis that it is 'transitive' or 'intransitive'. No noun should be manipulated according to its referent, e.g. whether it is human, or an animate, or an inanimate object. In the case that certain verbs or certain nouns do have a formal linguistic character - this character should be formulated in the program. If that is impossible then the group of words should be specified in a pseudo data list within the program.

While writing his program the linguist might discover some formal characteristics relating to topics which are traditionally dealt with by semantics. They may be treated therefore in his grammatical program. In other words, writing grammar as a program might show that some parts of the semantic meta-language can be reformulated by the grammar. The accurateness which should be used in programming does not diminish the realm of grammar. On the contrary, it enlarges it.

This last point is very interesting. The problem of the borderline between grammar and semantics is not easy to solve. The computer can play an important part in doing it. If the use of computers as a 'Handbook of Grammar' were more widespread we might assume that eventually the whole grammar of certain languages will be written as programs which will not contain, of course, any semantic consideration. When we achieve this situation these programs will give an 'applied definition' of the borderline between grammar and semantics. Whatever a linguist cannot include in his program is excluded from his grammar and should be considered as part of semantics. There is no question that the more sophisticated the linguist, the more semantics may be 'conquered' by the grammar. Looking at the problem in this way there is no need to emphasize how important it is for a linguist to write his programs himself.

NOTES

(1) e.g. Abraham S. Halkin, 201 Hebrew Verbs, conjugated in all the forms, 1970.

(2) This is a shortened version of a new description of Hebrew Morphology which will be published soon in Hebrew.

(3) I have shown these ways in a Hebrew paper, ''al ycirat šorašim xadašim etc.', Lešonenu la'am, no. 269 (1976), pp. 254-267. As a background see my 'Verb Formation', Encyclopaedia Judaica, vol. 8, pp. 124-140 (1971).

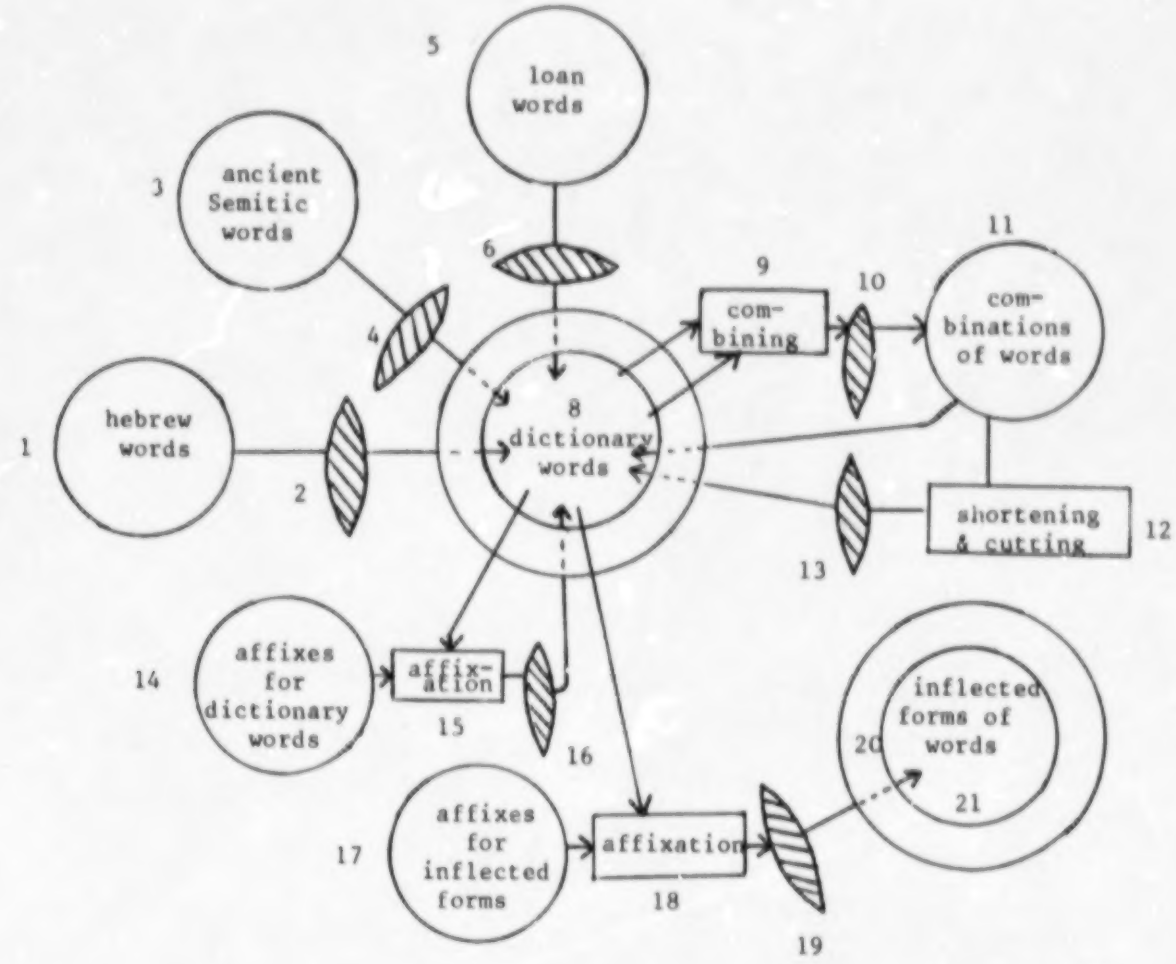
(4) An excellent illuminating description: S.R. Driver, A treatise on the Use of the Tenses in Hebrew, 1892, Appendix V, pp. 264-274. See also my Hebrew book Ha-mišpaṭ ha-pašuṭ, 1973, ch. 7.

(5) Several characters have been replaced here for technical reasons: ' is replaced by ', ^ is replaced by &, $ is replaced by š.

(6) 'ET is a marker for a direct object which follows.

(7) see Proceedings of the National Conference on Data Processing, Haifa, 1976 of IPA (Information Processing Association of Israel), pp. 431-440.

I would like to thank Prof. K.F. Bowden, Chairman, Department of Computer Science, University of Essex, England, and the faculty, who encouraged me to enter into the art when I stayed with them for one term.



the rings nos. 7 and 20 indicate socio-linguistic controls

— grammatical procedure

— a list of words or affixes

— phonetic or morphophonetic rules of realization

Figure A

Figure B

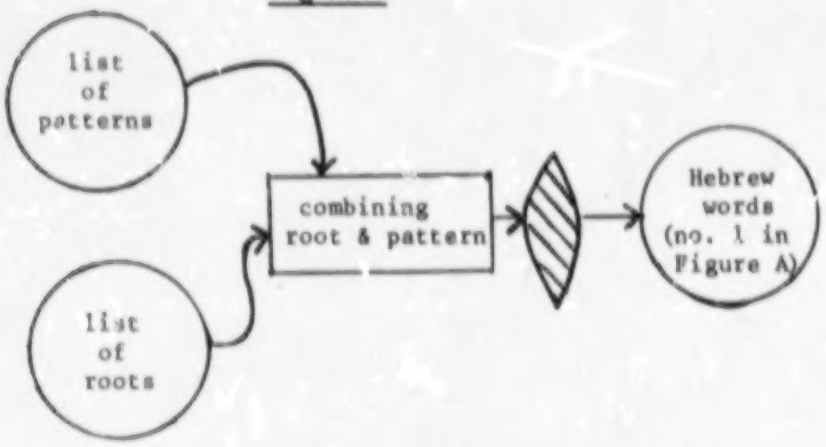
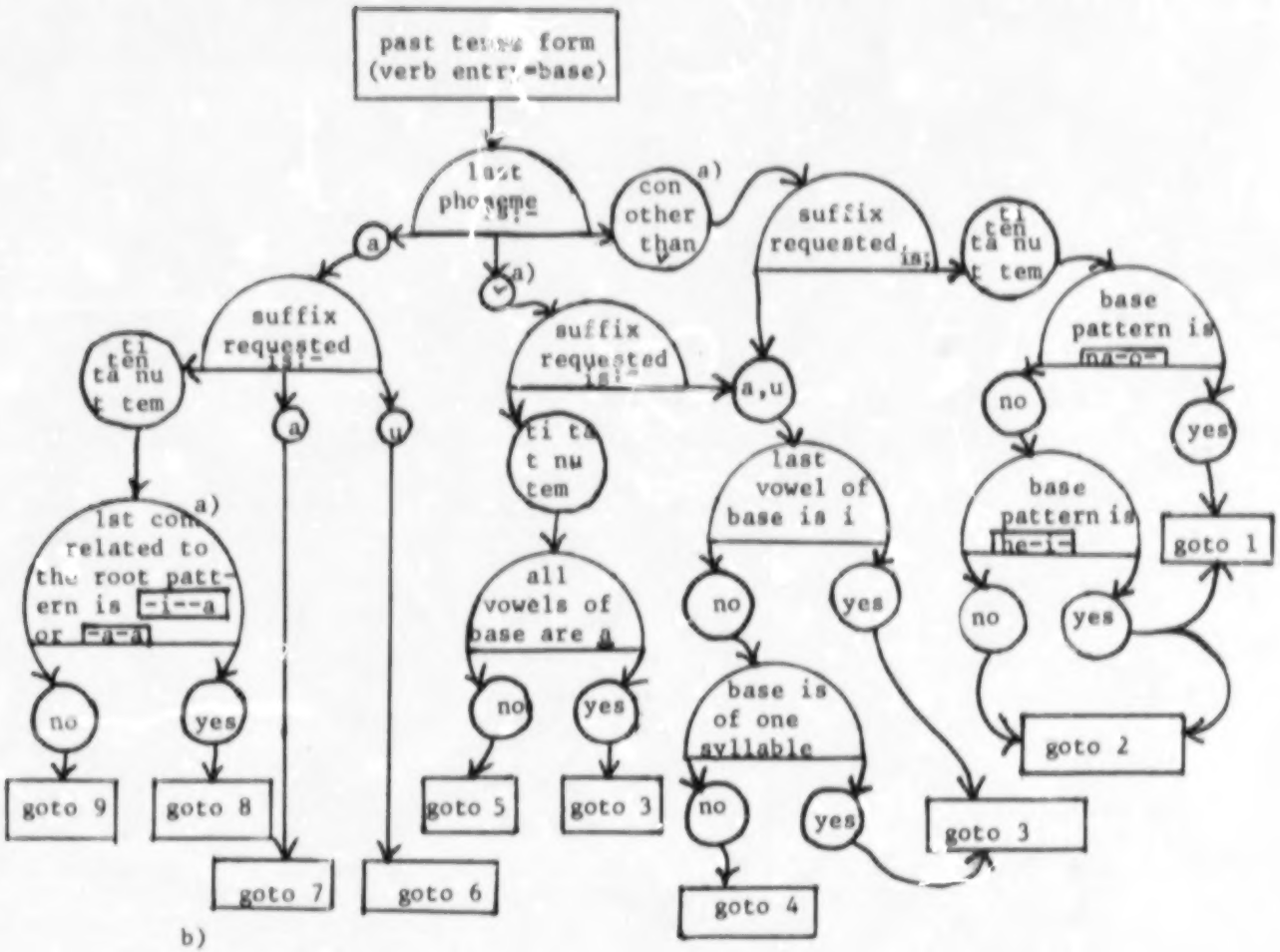


Figure C



1) b) O is inserted between the base and the suffix. If last vowel of the base is O, it becomes U. First vowel of the base is omitted. Go to 10.

2) b) Last vowel of the base, provided it is not O, must be a. Go to 3.

3) b) When a suffix is TEM, TEN, and the base is of two syllables and the first one of them is open, a of the first syllable is omitted and you go to 10. The same applies to e in pattern he-i-. Otherwise, you have got the requested form.

4) Last vowel of the base is omitted. Go to 10, then go to 11.

5) Last vowel of the base will be e.

6) Last vowel of the base is omitted.

7) Last vowel of the base is omitted. A ' is inserted before the suffix. Go to 10.

8) Last vowel of the base becomes i. Go to 3.

9) Last vowel of the base becomes ei.

10) If the omitted vowel is preceded by a guttural, an a (-:) replaces it.

11) If the omitted vowel is preceded by a series of two consonants, or if it is surrounded by two tokens of the same (or similiar) consonant, an e (:) replaces it (unless an a has already replaced it, according to 10).

---

a) is a sign for the glottal stop (ʔ). 'con' = consonant.

b) furtive pattax is omitted (furtive pattax is an additional a inserted before a guttural at the end of the word. e.g. heziac instead of hezix in the pattern he-i-).